INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-04-16 13:45 UTC

AgingFly Malware Compiles Its Own Weapons at Runtime, A Detection Engineering Challenge Targeting Ukraine's Critical Sectors

THREAT CAMPAIGN | **CRITICAL** | CVSS 7.5

SCC Item ID	SCC-CAM-2026-0179
Type	Threat Campaign
Severity	CRITICAL
CVSS Base Score	7.5
Affected Products	Windows systems broadly; Chromium-based browsers (Google Chrome, Microsoft Edge, Brave); WhatsApp for Windows
Published	2026-04-15T17:57:17
Discovery Source	Rss

Executive Summary

CERT-UA has identified an active intrusion campaign by threat cluster UAC-0247 deploying novel malware called AgingFly against Ukrainian government agencies, hospitals, and defense-affiliated organizations. The malware compiles its own attack code at runtime, making it largely invisible to traditional antivirus and signature-based security tools. Organizations supporting Ukrainian critical infrastructure or sharing networks with Ukrainian entities face elevated risk of credential theft, lateral movement, and sustained access by a persistent adversary.

Technical Analysis

AgingFly is a C# malware family attributed to UAC-0247 and reported by CERT-UA. Its defining characteristic is runtime compilation: the malware receives raw C# source code from its C2 server and compiles functional command handlers dynamically using the .NET compilation pipeline (CIL/Roslyn). This defeats static analysis and signature-based detection that inspects pre-built binaries. The campaign targets Windows systems broadly, with dedicated credential harvesting from Chromium-based browsers (Chrome, Edge, Brave) via T1555.003 and WhatsApp for Windows. Delivery involves a multi-stage chain consistent with spearphishing via link or attachment (T1566.002). Post-compromise tradecraft includes living-off-the-land lateral movement (T1218), process injection (T1055), reflective code loading (T1620), persistence via registry run keys (T1547.001), and C2 communication over standard web protocols (T1071.001) with possible protocol tunneling (T1572) and web service intermediaries (T1102). Relevant CWE mappings: CWE-494 (download of code without integrity check,

high confidence, core to runtime compilation design), CWE-312 (cleartext storage of sensitive information, high confidence, applies to credential harvesting from browser stores). No CVE is assigned. CVSS base 7.5 reflects campaign-level severity assessment. Source: BleepingComputer reporting on CERT-UA advisory (T3 secondary source); direct CERT-UA advisory confirmation recommended before operationalizing IOCs.

Action Checklist

1. Containment. Identify and isolate Windows endpoints in or connected to Ukrainian government, healthcare, or defense-affiliated networks. Block outbound .NET compiler process (csc.exe, vbc.exe, msbuild.exe) spawning from unexpected parent processes. Restrict or monitor network egress from Chromium-based browser profile directories and WhatsApp for Windows data paths.
2. Detection. Hunt for anomalous invocations of csc.exe, vbc.exe, or MSBuild.exe spawned by non-development parent processes. Review Windows Security event logs for process creation (Event ID 4688) involving .NET compiler binaries. Monitor for outbound connections from .NET runtime processes to non-whitelisted C2 infrastructure. Check EDR telemetry for T1055 (process injection), T1620 (reflective code loading), and T1218 (signed binary proxy execution) chains. Review browser credential store access logs where available. Confirm IOCs against direct CERT-UA advisory before deploying to production detection rules.
3. Eradication. Remove AgingFly persistence mechanisms: audit and clean registry run keys (HKCU\Software\Microsoft\Windows\CurrentVersion\Run and HKLM equivalent) for unknown entries. Terminate and remove any identified AgingFly-related processes and dropped files. Rotate all credentials stored in Chromium-based browsers and WhatsApp on affected systems. Revoke and reissue authentication tokens for accounts accessed from compromised endpoints.
4. Recovery. Validate that no .NET compiler processes are spawning from unexpected parents post-remediation. Restore affected systems from known-good backups taken prior to initial compromise where forensic analysis indicates deep persistence. Reimage endpoints where full scope of access cannot be confirmed. Monitor re-introduced systems for recurrence of IOC patterns for a minimum of 30 days.
5. Post-Incident. Assess gaps in behavioral detection coverage for runtime code compilation chains. Evaluate whether application control policies (e.g., AppLocker, WDAC) are configured to block unauthorized invocations of csc.exe and MSBuild.exe in non-developer environments. Review credential storage hygiene policies for browser-stored passwords, enforce password manager controls that do not rely on browser native stores. Map control gaps to NIST CSF DE.CM (continuous monitoring) and PR.AC (access control) domains.

IR / Forensic Enrichment

Triage Priority	IMMEDIATE
Escalation Criteria	Escalate immediately to senior leadership, legal, and any applicable national CERT (CERT-UA liaison if applicable) if: confirmed credential exfiltration from Chromium browsers or WhatsApp is identified on endpoints with access to government, healthcare, or defense systems (triggering breach notification obligations under HIPAA, NIS2, or applicable national law), if lateral movement indicators (Event ID 4624 Type 3 logons from compromised hosts) are detected beyond the initially identified endpoints, or if forensic analysis cannot bound the scope of AgingFly's runtime-compiled payload activity within 24 hours of detection.

Recovery Notes	Prioritize reimage over in-place remediation for any endpoint where forensic analysis cannot confirm the full set of AgingFly-compiled payloads — AgingFly's runtime compilation model means the full attack surface of dropped code may not be reconstructable from artifacts alone. Restore from backups dated before the earliest identified Sysmon or Event ID 4688 anomaly for csc.exe or msbuild.exe, not simply before the first alert, as AgingFly may have established persistence silently before triggering behavioral indicators. Maintain active Sysmon monitoring for .NET compiler process trees and outbound connections from .NET runtime processes for a minimum of 30 days post-reintroduction, as UAC-0247 campaigns targeting Ukrainian critical sectors have demonstrated persistence reestablishment following incomplete remediation.
Forensic Artifacts	%TEMP% and %APPDATA% transient .cs, .vb, .dll, and .exe files — AgingFly's runtime compilation model generates temporary source code and compiled output in user-writable directories; file creation timestamps in these paths during the intrusion window directly evidence the compilation chain and may contain decompilable payload logic. | Sysmon Event ID 1 (Process Create) logs showing csc.exe, vbc.exe, or msbuild.exe with ParentImage values outside of Visual Studio (devenv.exe), dotnet CLI, or approved CI/CD agent processes — the parent-child relationship is the primary behavioral fingerprint of AgingFly's living-off-the-land compilation technique (MITRE T1127.001). | Chromium browser SQLite credential stores at %LOCALAPPDATA%\Google\Chrome\User Data\Default\Login Data, %LOCALAPPDATA%\Microsoft\Edge\User Data\Default\Login Data, and %LOCALAPPDATA%\BraveSoftware\Brave-Browser\User Data\Default\Login Data — file last-accessed timestamps and Windows Security Event ID 4663 object access records showing non-browser processes reading these files confirm UAC-0247 credential harvesting activity. | WhatsApp for Windows session and local storage data at %APPDATA%\WhatsApp\Local Storage\leveldb\ and %APPDATA%\WhatsApp\IndexedDB\ — access or modification of these paths by processes other than WhatsApp.exe (identifiable via Sysmon Event ID 10, Process Access, SourceImage != WhatsApp.exe) constitutes evidence of WhatsApp credential or session token harvesting by AgingFly. | Windows Security Event ID 4688 (Process Creation) with full command-line logging enabled — specifically entries where ProcessName matches csc.exe and CommandLine contains references to /tmp, %TEMP%, or dynamically named .cs files, which represent the in-memory-to-disk compilation artifacts unique to AgingFly's runtime weaponization approach and distinguish it from legitimate developer activity.

Per-Action IR Details

Containment — Identify and isolate Windows endpoints in or connected to Ukrainian government, healthcare, or defense-adjacent networks. Block outbound .NET compiler process (csc.exe, vbc.exe, msbuild.exe) spawning from unexpected parent processes. Restrict or monitor network egress from Chromium-based browser profile directories and WhatsApp for Windows data paths.

NIST Phase: Containment

Reference: NIST 800-61r3 §3.3 — Containment Strategy

Controls: NIST IR-4 (Incident Handling), NIST SI-4 (System Monitoring), NIST SC-7 (Boundary Protection), CIS 4.4 (Implement and Manage a Firewall on Servers), CIS 4.5 (Implement and Manage a Firewall on End-User Devices), CIS 13.4 (Perform Traffic Filtering Between Network Segments)

Compensating: Use Windows Defender Firewall with Advanced Security to create outbound block rules targeting csc.exe (%WINDIR%\Microsoft.NET\Framework*\csc.exe), vbc.exe, and msbuild.exe via: `netsh advfirewall firewall add rule name='Block CSC Outbound' dir=out program='C:\Windows\Microsoft.NET\Framework64\v4.0.30319\csc.exe' action=block`. Deploy Sysmon with a configuration that logs Event ID 1 (Process Create) with parent-process tracking to immediately surface unexpected spawning chains. For network egress from browser data paths (%LOCALAPPDATA%\Google\Chrome\User Data\, %APPDATA%\WhatsApp\), use Windows Defender Firewall

application rules to block non-browser executables from those directories, or use osquery query `SELECT pid, name, path FROM processes WHERE path LIKE '%Chrome%User Data%' OR path LIKE '%WhatsApp%'` to identify anomalous process access.

Evidence: Before isolating endpoints, capture: (1) Sysmon Event ID 1 logs showing csc.exe, vbc.exe, or msbuild.exe process trees — specifically parent process names, command-line arguments, and working directories to establish AgingFly's compilation invocation chain. (2) Network connection state via `netstat -ano` and map PIDs to processes — look for .NET runtime processes (dotnet.exe, csc.exe) with established outbound connections to non-Microsoft, non-CDN IP ranges. (3) Browser profile credential store files: %LOCALAPPDATA%\Google\Chrome\User Data\Default\Login Data, %LOCALAPPDATA%\Microsoft\Edge\User Data\Default\Login Data, and %APPDATA%\Brave Software\Brave-Browser\User Data\Default\Login Data — hash these files before isolation to document their state. (4) WhatsApp for Windows credential and session data at %APPDATA%\WhatsApp\Local Storage\ and %APPDATA%\WhatsApp\IndexedDB\. (5) Memory dump of any active .NET compiler or runtime processes using ProcDump (`procdump -ma `) to capture in-memory compiled payloads before power-down.

Detection — Hunt for anomalous invocations of csc.exe, vbc.exe, or MSBuild.exe spawned by non-development parent processes. Review Windows Security event logs for process creation (Event ID 4688) involving .NET compiler binaries. Monitor for outbound connections from .NET runtime processes to non-whitelisted C2 infrastructure. Check EDR telemetry for T1055 (process injection), T1620 (reflective code loading), and T1218 (signed binary proxy execution) chains. Review browser credential store access logs where available. Confirm IOCs against direct CERT-UA advisory before deploying to production detection rules.

NIST Phase: Detection Analysis

Reference: NIST 800-61r3 §3.2 — Detection and Analysis

Controls: NIST SI-4 (System Monitoring), NIST AU-2 (Event Logging), NIST AU-6 (Audit Record Review, Analysis, and Reporting), NIST AU-12 (Audit Record Generation), CIS 8.2 (Collect Audit Logs), CIS 7.1 (Establish and Maintain a Vulnerability Management Process)

Compensating: Enable Process Creation Auditing via Group Policy (Computer Configuration > Windows Settings > Security Settings > Advanced Audit Policy > Detailed Tracking > Audit Process Creation = Success) to populate Event ID 4688 with command-line arguments. Run this PowerShell query across collected Security logs to surface AgingFly compiler activity: `Get-WinEvent -LogName Security | Where-Object {$_.Id -eq 4688} | Where-Object {$_.Message -match 'csc.exe|vbc.exe|msbuild.exe'} | Select-Object TimeCreated, @{N='CommandLine';E={$_.Properties[8].Value}}, @{N='ParentProcess';E={$_.Properties[13].Value}} | Export-Csv compiler_hits.csv`. Deploy the public Sigma rule for 'Suspicious MSBuild Execution' (SigmaHQ ruleset, tag: process_creation) converted to Windows Event Log format. For T1620 reflective code loading, use Sysmon Event ID 7 (Image Loaded) filtered on unsigned or low-reputation DLLs loaded into .NET runtime processes. For browser credential store access hunting, use Sysmon Event ID 10 (Process Access) where TargetImage matches chrome.exe or msedge.exe but SourceImage is not the browser itself.

Evidence: Before concluding detection scope: (1) Export Windows Security Event Log filtered for Event ID 4688 with ProcessName containing csc.exe, vbc.exe, or msbuild.exe — retain the full command-line field which will contain AgingFly's runtime-generated source code paths or temp directory references (e.g., %TEMP%*.cs). (2) Collect Sysmon Event ID 3 (Network Connection) logs for processes named csc.exe or dotnet.exe to identify C2 IP addresses and domains specific to UAC-0247 infrastructure. (3) Pull Windows Security Event ID 4663 (Object Access) for reads against Chrome, Edge, and Brave Login Data SQLite files — these will show which process accessed stored credentials, confirming browser credential harvesting. (4) Review %TEMP% and %APPDATA% directories for transient .cs, .vb, or .dll files that AgingFly's runtime compilation would generate and potentially not fully clean up. (5) Cross-reference identified IPs and domains against the current CERT-UA UAC-0247 advisory IOC list before treating as confirmed — do not assume training-data IOCs are current.

Eradication — Remove AgingFly persistence mechanisms: audit and clean registry run keys (HKCU\Software\Microsoft\Windows\CurrentVersion\Run and HKLM equivalent) for unknown entries. Terminate and remove any identified AgingFly-related processes and dropped files. Rotate all credentials stored in Chromium-based browsers and WhatsApp on affected systems. Revoke and reissue authentication

tokens for accounts accessed from compromised endpoints.

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication

Controls: NIST IR-4 (Incident Handling), NIST SI-2 (Flaw Remediation), NIST SI-7 (Software, Firmware, and Information Integrity), NIST AC-2 (Account Management), CIS 5.3 (Disable Dormant Accounts), CIS 4.7 (Manage Default Accounts on Enterprise Assets and Software)

Compensating: Export full registry hive before modification: `reg export HKCU\Software\Microsoft\Windows\CurrentVersion\Run pre_clean_HKCU_run.reg` and `reg export HKLM\Software\Microsoft\Windows\CurrentVersion\Run pre_clean_HKLM_run.reg`. Use Autoruns (Sysinternals, free) with VirusTotal integration enabled — scan all autorun locations and flag entries pointing to %TEMP%, %APPDATA%, or unsigned binaries. For credential rotation without an enterprise IAM: use the browser's built-in password export (chrome://settings/passwords), then delete the Login Data file and force re-authentication on all saved sites. For WhatsApp session revocation, navigate to WhatsApp Settings > Linked Devices and remove all linked device sessions. For token revocation on Microsoft 365 or Azure AD accounts accessed from compromised hosts, use: `Revoke-AzureADUserAllRefreshToken -ObjectId ` (requires AzureAD PowerShell module, free).

Evidence: Before removing any artifact: (1) Export registry run keys in full before cleaning — `reg export HKCU\Software\Microsoft\Windows\CurrentVersion\Run evidence_HKCU.reg` — AgingFly persistence entries will likely point to compiled executables in non-standard paths like %TEMP% or %APPDATA%\Roaming subdirectories. (2) Hash all identified AgingFly-related files with `Get-FileHash -Algorithm SHA256 ` and submit hashes to CERT-UA or VirusTotal for confirmation before deletion. (3) Capture the full contents of %TEMP% and any discovered staging directories — AgingFly's runtime compilation leaves temporary source (.cs/.vb) and compiled output (.dll/.exe) files that constitute direct forensic evidence of the compilation chain. (4) Document all entries from Chromium browser Login Data files (read-only SQLite query: `sqlite3 'Login Data' 'SELECT origin_url, username_value, date_created FROM logins'`) to establish the scope of potentially compromised credentials before rotation. (5) Export Windows Security Event ID 4624/4625 (Logon Success/Failure) for accounts active on the compromised endpoint during the suspected intrusion window to identify lateral movement or credential reuse.

Recovery — Validate that no .NET compiler processes are spawning from unexpected parents post-remediation. Restore affected systems from known-good backups taken prior to initial compromise where forensic analysis indicates deep persistence. Reimage endpoints where full scope of access cannot be confirmed. Monitor re-introduced systems for recurrence of IOC patterns for a minimum of 30 days.

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Recovery

Controls: NIST IR-4 (Incident Handling), NIST CP-9 (System Backup), NIST CP-10 (System Recovery and Reconstitution), NIST SI-4 (System Monitoring), CIS 7.3 (Perform Automated Operating System Patch Management), CIS 11.2 (Perform Automated Backups)

Compensating: Deploy Sysmon on all restored/reimaged endpoints before reconnecting to the network, configured with a rule that generates Event ID 1 alerts for csc.exe, vbc.exe, or msbuild.exe with parent processes outside of Visual Studio, dotnet CLI, or approved build system paths. For the 30-day monitoring period, use a scheduled PowerShell task (run every 4 hours) to query Event ID 4688 for compiler binary invocations and write results to a monitored log share: `Get-WinEvent -LogName Security -MaxEvents 5000 | Where-Object {$_.Id -eq 4688 -and $_.Message -match 'csc.exe|vbc.exe|msbuild.exe'} | Export-Csv \\logserver\agingfly_watch_$(Get-Date -Format yyyyMMdd_HHmm).csv`. Validate backup integrity before restore by checking backup hash against stored pre-incident baseline using `Get-FileHash` on the backup image.

Evidence: Before marking recovery complete: (1) Run a full Autoruns scan on the restored system and diff the output against a known-clean baseline — any entries that reappear after remediation indicate a persistence mechanism that survived cleanup, potentially in locations beyond the standard Run keys (e.g., scheduled tasks, WMI subscriptions, service registrations). (2) Capture a post-remediation Sysmon Event ID 1 baseline for 24 hours on restored systems to confirm absence of csc.exe, vbc.exe, or msbuild.exe anomalous invocations before declaring the system clean. (3) Verify browser credential stores (Chrome, Edge, Brave Login Data files) have been replaced with empty post-rotation versions by checking file modification timestamps and size — a non-empty Login Data file on a freshly restored system

that was supposed to have credentials cleared is an indicator of incomplete eradication. (4) Confirm no outbound connections from .NET runtime processes using `netstat -ano` on the restored system and correlate PIDs to process names via `tasklist /fi 'PID eq '` for any established connections outside expected Microsoft update and telemetry ranges.

Post-Incident — Assess gaps in behavioral detection coverage for runtime code compilation chains. Evaluate whether application control policies (e.g., AppLocker, WDAC) are configured to block unauthorized invocations of csc.exe and MSBuild.exe in non-developer environments. Review credential storage hygiene policies for browser-stored passwords — enforce password manager controls that do not rely on browser native stores. Map control gaps to NIST CSF DE.CM (continuous monitoring) and PR.AC (access control) domains.

NIST Phase: Post Incident

Reference: NIST 800-61r3 §4 — Post-Incident Activity

Controls: NIST IR-4 (Incident Handling), NIST IR-8 (Incident Response Plan), NIST SI-4 (System Monitoring), NIST AC-6 (Least Privilege), NIST CM-7 (Least Functionality), CIS 2.3 (Address Unauthorized Software), CIS 4.6 (Securely Manage Enterprise Assets and Software), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 8.2 (Collect Audit Logs)

Compensating: For AppLocker (available in Windows Enterprise/Education, free with OS license): create a Deny rule for %WINDIR%\Microsoft.NET\Framework*\csc.exe and %WINDIR%\Microsoft.NET\Framework*\vbc.exe for all user groups except a designated 'Developers' group via Group Policy (Computer Configuration > Windows Settings > Security Settings > Application Control Policies > AppLocker). For WDAC on Windows 10/11 Pro and above: generate a policy using `New-CIPolicy` that sets csc.exe and msbuild.exe as blocked in non-developer OUs. For browser credential hygiene without enterprise budget: enforce via Group Policy that Chrome's built-in password manager is disabled (`PasswordManagerEnabled = false` under Computer Configuration > Administrative Templates > Google > Google Chrome > Password Manager) and mandate use of a free offline password manager such as KeePassXC. Document the AgingFly runtime compilation technique in the internal threat library and create a Sigma rule (publish to SigmaHQ community) for csc.exe spawned by non-IDE parent processes to benefit the broader defender community.

Evidence: Post-incident review should analyze: (1) Whether existing Sysmon or Windows Event Log collection was capturing command-line arguments (Event ID 4688 CommandLine field requires 'Include command line in process creation events' audit policy to be enabled) — if not, this gap allowed AgingFly's compilation invocations to go undetected. (2) Gap analysis on AppLocker or WDAC policy coverage — pull current policy XML (`Get-AppLockerPolicy -Effective -Xml`) and confirm whether csc.exe, vbc.exe, and msbuild.exe are explicitly addressed or fell through default allow rules. (3) Review browser password manager usage across the estate via Group Policy reporting to determine how many users had credentials stored in Chromium native stores at the time of the incident, establishing the true credential exposure scope for UAC-0247. (4) Correlate the incident timeline against CERT-UA advisory publication dates to measure detection gap — the delta between when UAC-0247 activity began and when your environment first generated an alert is your mean time to detect (MTTD) baseline for this threat class.

Detection Guidance

Primary behavioral indicator: csc.exe, vbc.exe, or MSBuild.exe spawned by an unexpected parent process (e.g., a browser, Office application, or unknown executable). Windows Security Event ID 4688 with process creation command-line logging enabled will surface this if command-line auditing is active. Sysmon Event ID 1 (ProcessCreate) with ParentImage not matching approved development toolchains is a high-fidelity detection opportunity. Secondary indicators: outbound HTTP/HTTPS connections originating from .NET runtime processes (clrjit.dll, clr.dll loaded in non-standard host processes); anomalous reads of Chromium browser profile paths (e.g., %LOCALAPPDATA%\Google\Chrome\User Data\Default\Login Data) by non-browser processes (T1555.003); and registry write activity to HKCU\Software\Microsoft\Windows\CurrentVersion\Run from non-administrative processes (T1547.001). For network detection, monitor for encoded or compressed

data exfiltration patterns consistent with T1132 (data encoding) and T1041 (exfil over C2 channel). MITRE ATT&CK techniques of highest detection value for this campaign: T1620 (reflective code loading), T1027.010 (command obfuscation), T1555.003 (credentials from web browsers), T1547.001 (registry run keys). Note: specific IOCs (hashes, IPs, domains) should be pulled directly from the CERT-UA advisory; the BleepingComputer source cited is T3 tier and IOCs from it require confirmation before production deployment.

Indicators of Compromise

Type	Value	Context	Confidence
DOMAIN	`[Pending direct CERT-UA advisory confirmation]`	C2 infrastructure used by UAC-0247 for AgingFly runtime code delivery. Specific IOCs not independently verified from T3 source; retrieve from direct CERT-UA advisory before operationalizing.	LOW
HASH	`[Pending direct CERT-UA advisory confirmation]`	AgingFly dropper or loader hashes. Not extractable from T3 source at sufficient confidence; retrieve from direct CERT-UA advisory.	LOW

Framework Mappings

MITRE-ATTACK

- **T1071.001** — Web Protocols
- **T1218** — System Binary Proxy Execution
- **T1046** — Network Service Discovery
- **T1132** — Data Encoding
- **T1041** — Exfiltration Over C2 Channel
- **T1027** — Obfuscated Files or Information
- **T1572** — Protocol Tunneling
- **T1140** — Deobfuscate/Decode Files or Information
- **T1102** — Web Service
- **T1059.001** — PowerShell
- **T1036** — Masquerading
- **T1055** — Process Injection
- **T1620** — Reflective Code Loading
- **T1539** — Steal Web Session Cookie
- **T1105** — Ingress Tool Transfer
- **T1059.003** — Windows Command Shell
- **T1027.010** — Command Obfuscation
- **T1547.001** — Registry Run Keys / Startup Folder
- **T1566.002** — Spearphishing Link

- **T1555.003** — Credentials from Web Browsers

NIST-800-53R5

- **CA-7** — Continuous Monitoring
- **SC-7** — Boundary Protection
- **SI-4** — System Monitoring
- **SI-3** — Malicious Code Protection
- **CM-7** — Least Functionality
- **SI-7** — Software, Firmware, and Information Integrity
- **AC-6** — Least Privilege
- **AT-2** — Literacy Training and Awareness
- **SI-8** — Spam Protection
- **CM-3** — Configuration Change Control
- **SI-10** — Information Input Validation

OWASP-TOP10-2021

- **A08:2021** — Software and Data Integrity Failures
- **A03:2021** — Injection

CIS-V8

- **2.5** — Allowlist Authorized Software
- **2.6** — Allowlist Authorized Libraries
- **16.10** — Apply Secure Design Principles in Application Architectures
- **6.3** — Require MFA for Externally-Exposed Applications
- **8.2** — Collect Audit Logs

ISO-27001-2022

- **A.8.28** — Secure coding

HIPAA-SECURITY

- **164.312(d)** — Person or Entity Authentication

SOC2-TSC

- **CC6.1** — Logical access security software, infrastructure, and architectures

NIST-CSF-2

- **DE.CM-01** — Networks and network services are monitored

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1071.001	Web Protocols	Command-And-Control

Technique ID	Technique Name	Tactic
T1218	System Binary Proxy Execution	Defense-Evasion
T1046	Network Service Discovery	Discovery
T1132	Data Encoding	Command-And-Control
T1041	Exfiltration Over C2 Channel	Exfiltration
T1027	Obfuscated Files or Information	Defense-Evasion
T1572	Protocol Tunneling	Command-And-Control
T1140	Deobfuscate/Decode Files or Information	Defense-Evasion
T1102	Web Service	Command-And-Control
T1059.001	PowerShell	Execution
T1036	Masquerading	Defense-Evasion
T1055	Process Injection	Defense-Evasion
T1620	Reflective Code Loading	Defense-Evasion
T1539	Steal Web Session Cookie	Credential-Access
T1105	Ingress Tool Transfer	Command-And-Control
T1059.003	Windows Command Shell	Execution
T1027.010	Command Obfuscation	Defense-Evasion
T1547.001	Registry Run Keys / Startup Folder	Persistence
T1566.002	Spearphishing Link	Initial-Access
T1555.003	Credentials from Web Browsers	Credential-Access

Sources

Source	URL	Tier
Security News	https://www.bleepingcomputer.com/news/security/new-agingfly-malware...	T3
Chrome and Chromium-based browsers receive fixes for exploited ...	https://fieldeffect.com/blog/chrome-chromium-browsers-fixes-exploit...	T3

Source	URL	Tier
Chromium Based Browsers: The Complete Guide for 2026	https://undetectable.io/blog/chromium-based-browsers/	T3
Print - ■Vulnerability Alert■Chromium-based browsers ...	https://lis.mcut.edu.tw/p/16-1013-77554.php?Lang=en	T3
Chrome users have been warned about multiple security issues by ...	https://www.facebook.com/cnnnews18/posts/chrome-users-have-been-war...	T3

Generated 2026-04-16 13:45 UTC by TJS Security Command Center